CS 6301

Lecture 2. **Text** Classification

Slides adapted from Rui Zhang, Graham Neubig

Outline - Key Concepts

NLP

- Text Classification
- Language Modeling
- Word Representations/Embedding

ML

- Discriminative Model vs Generative Model
- Objective Function
- Gradient Descent
- Evaluation
- Statistical Testing
- Feed-forward Neural Networks and Recurrent Neural Networks

Text Classification

Input X	Output Y	Task
Text	Label	Text Classification
(e.g., Sentiment Analysis)		
Text	Linguistic Structure	Structured Prediction (e.g., POS Tagging)
Text	Text	Text Generation
(e.g., Translation)		

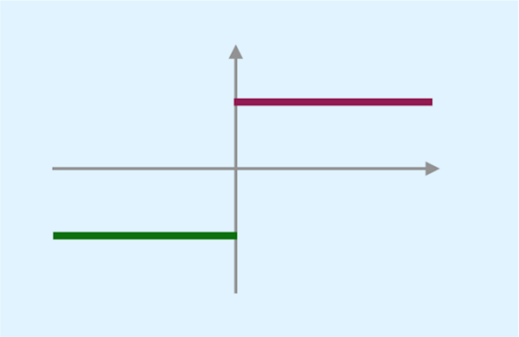
The/**DT** planet/**NN** Jupiter/**NNP** and/**CC** its/**PPS** moons/**NNS** are/**VBP** in/**IN** effect/**NN** a/**DT** mini-solar/**JJ** system/**NN** ./**.**

Text Classification

Input X	Output Y	Task
Text	Label	Text Classification (e.g., Sentiment Analysis)
Text	Linguistic Structure	Structured Prediction (e.g., POS Tagging)
Text	Text	Text Generation (e.g., Translation)

Discriminative Model: Calculate the conditional probability distribution of class labels Y given the input data X.

From Prediction Score to Probability

$\boldsymbol{w}^\top \boldsymbol{x} \longrightarrow$  $\longrightarrow$ $y = \begin{cases} +1 & \text{if } \boldsymbol{w}^\top \boldsymbol{x} \geq 0 \\ -1 & \text{if } \boldsymbol{w}^\top \boldsymbol{x} < 0 \end{cases}$

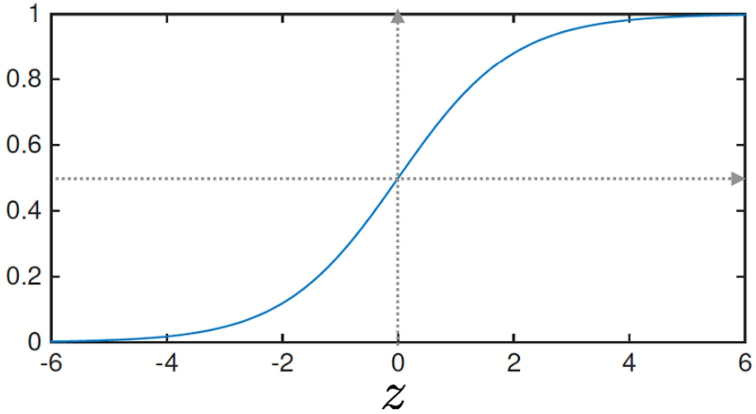
$\boldsymbol{w}^\top \boldsymbol{x} \longrightarrow$ **?** $\longrightarrow$

$\mathbb{P}[y = +1]$

$\mathbb{P}[y = -1] = 1 - \mathbb{P}[y = +1]$

Sigmoid for Binary Classification

$$\sigma(z) = \frac{1}{1+e^{-z}}$$

$\boldsymbol{w}^\top \boldsymbol{x}$ →  → $\mathbb{P}[y=+1 \mid \boldsymbol{x}; \boldsymbol{w}]$

$$\mathbb{P}\left[y=+1|\boldsymbol{x};\boldsymbol{w}\right] = \sigma(\boldsymbol{w}^\top \boldsymbol{x}) = \frac{1}{1+e^{-\boldsymbol{w}^\top \boldsymbol{x}}}$$

$$\mathbb{P}\left[y=-1|\boldsymbol{x};\boldsymbol{w}\right] = 1-\mathbb{P}\left[y=+1|\boldsymbol{x};\boldsymbol{w}\right] = \sigma(-\boldsymbol{w}^\top \boldsymbol{x}) = \frac{1}{1+e^{\boldsymbol{w}^\top \boldsymbol{x}}}$$

Softmax for Multi-class Classification

Softmax extends the idea of sigmoid into a multi-class classification.
It converts scores to a probability distribution of class labels.

The predicted probability for the j'th class given a sample vector $\mathbf{x}$ and a weighting vector $\mathbf{w}$ is

$$P(y=j \mid \mathbf{x}) = \frac{e^{\mathbf{x}^\mathsf{T}\mathbf{w}_j}}{\sum_{k=1}^{K} e^{\mathbf{x}^\mathsf{T}\mathbf{w}_k}}$$

Softmax Example

$$P(y=j \mid \mathbf{x}) = \frac{e^{\mathbf{x}^\mathsf{T}\mathbf{w}_j}}{\sum_{k=1}^{K} e^{\mathbf{x}^\mathsf{T}\mathbf{w}_k}}$$

$$\begin{bmatrix} 8 \\ 5 \\ 0 \end{bmatrix}$$

$$\sum_{j=1}^{K} e^{z_j} = e^{z_1} + e^{z_2} + e^{z_3} = 2981.0 + 148.4 + 1.0 = 3130.4$$

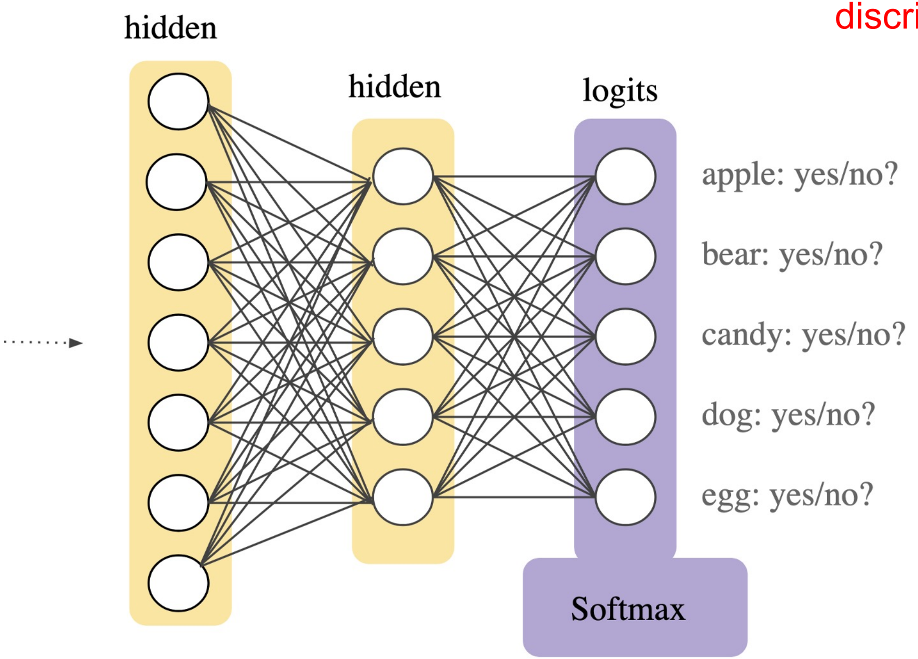
$$e^{z_1} = e^{8} = 2981.0$$
$$e^{z_2} = e^{5} = 148.4$$
$$e^{z_3} = e^{0} = 1.0$$

$$\frac{2981.0}{3130.4} = 0.9523$$
$$\frac{148.4}{3130.4} = 0.0474$$
$$\frac{1.0}{3130.4} = 0.0003$$

https://deepai.org/machine-learning-glossary-and-terms/softmax-layer

Softmax in Neural Networks

discriminate among the inputs



Class	Probability
apple	0.001
bear	0.04
candy	0.008
dog	0.95
egg	0.001

https://developers.google.com/machine-learning/crash-course/multi-class-neural-networks/softmax

Models

- **Discriminative Model**: Calculate the **conditional probability distribution** of class labels Y given the input data X.

- Joint (e.g. Naïve Bayes)
    - Parameters from data statistics
    - Parameters: probabilistic interpretation
    - Training: one pass through the data

- Perceptron
    - Parameters from reactions to mistakes
    - Parameters: discriminative interpretation
    - Training: go through the data until validation accuracy maxes out

Discriminative Model vs Generative Model

Discriminative Model: Calculate the conditional probability distribution of class labels Y given the input data X.

$$P(y|X)$$

Generative Model: Directly Model

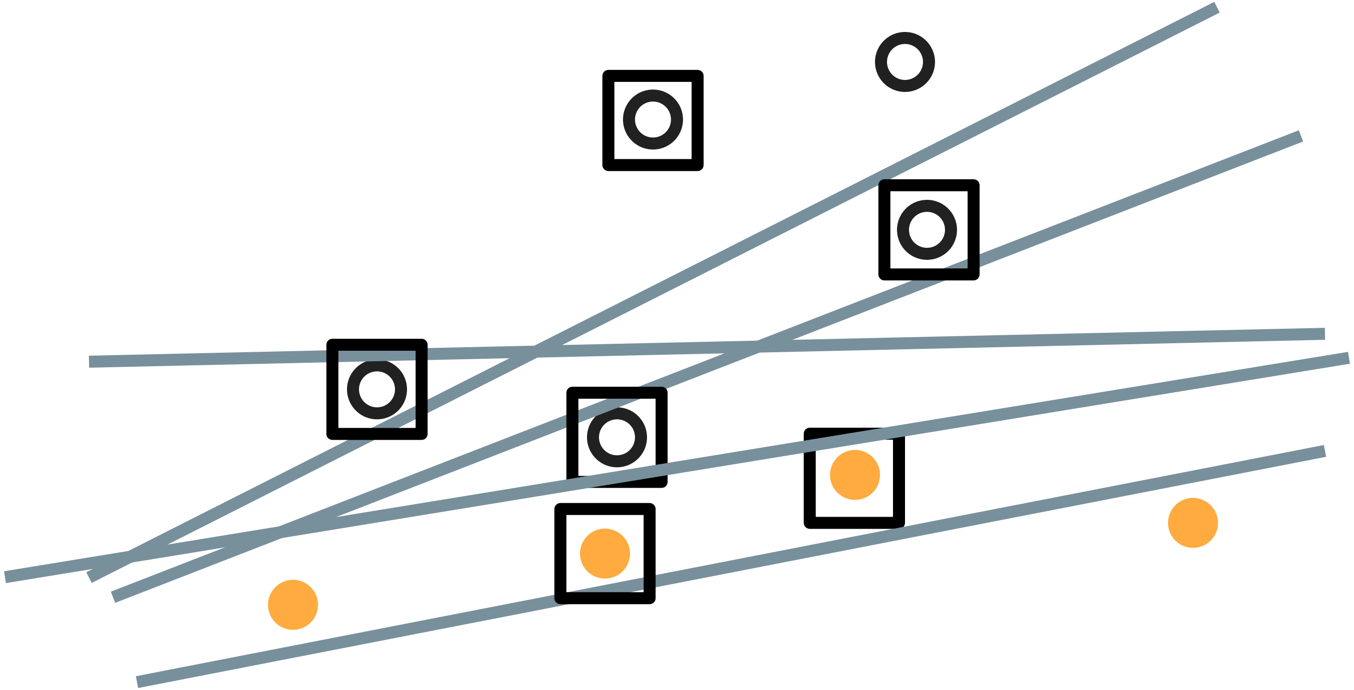
- maximize the joint probability of the training set of labeled documents
- maximum likelihood estimation (MLE)

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\, \mathrm{p}(\boldsymbol{x}^{(1:N)}, y^{(1:N)}; \boldsymbol{\theta})$$

Perceptron (Separable Case)

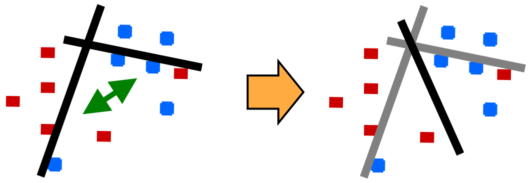
- The perceptron algorithm
  - Iteratively processes the training set, reacting to training errors
  - Can be thought of as trying to drive down training error
- The online (binary → $y = \pm 1$) perceptron algorithm:
  - Start with zero weights
  - Visit training instances $(X^{(i)}, y^{(i)})$ one by one, until all correct
    - Make a prediction
    - If correct ($y^* == y^{(i)}$): no change, goto next example!
    - If wrong: adjust weights

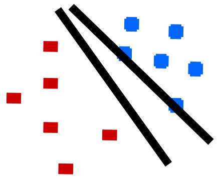
Perceptron (Separating Hyperplane)



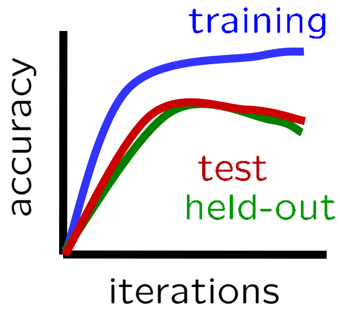
Problems with the Perceptron

- Noise: if the data isn't separable, weights might thrash
  - Averaging weight vectors over time can help (averaged perceptron)





- Mediocre generalization: finds a "barely" separating solution
- Overtraining: test / held-out accuracy usually rises, then falls
  - Overtraining is a kind of overfitting



Models

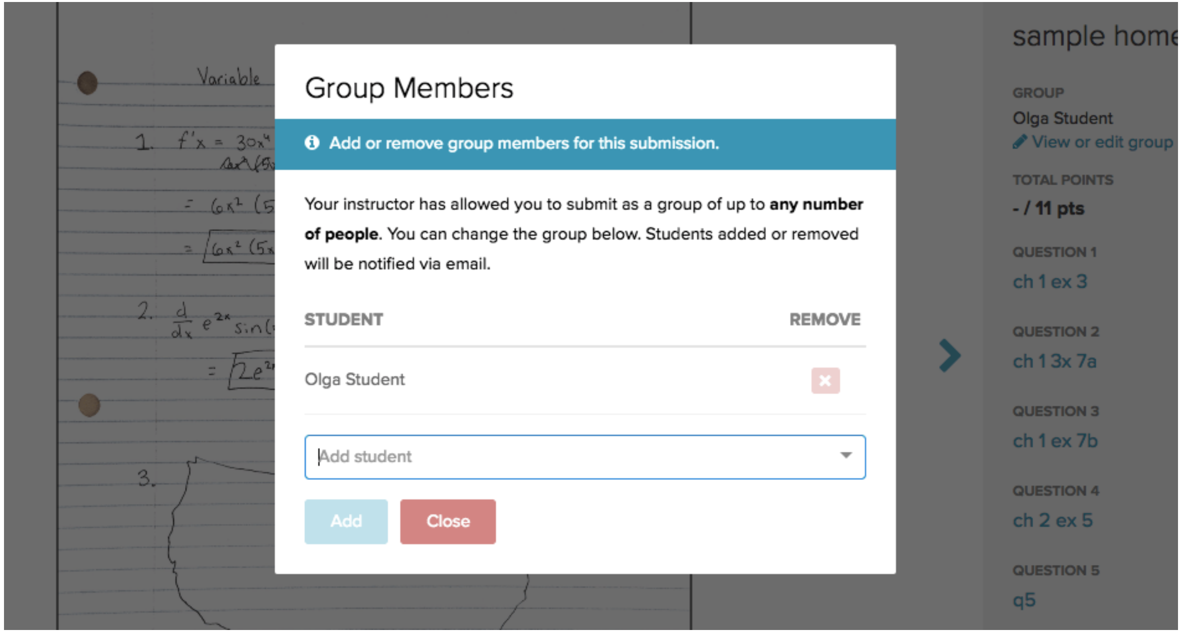
- **Discriminative Model**: Calculate the **conditional probability distribution** of class labels Y given the input data X. (Neural Network)

- Joint (e.g. Naïve Bayes)
  - Parameters from data statistics
  - Parameters: probabilistic interpretation
  - Training: one pass through the data

- Perceptron
  - Parameters from reactions to mistakes
  - Parameters: discriminative interpretation
  - Training: go through the data until validation accuracy maxes out

Logistics

- Gradescope submission
  - Assignments, Presentation slides, final project report



Discriminative Model

The discriminative model is parameterized by θ

$$P(y|X;\theta)$$

Discriminative Model Objective Function

The discriminative model is parameterized by θ

$$P(y|X;\theta)$$

We often use **negative log likelihood** over training data as our **objective function** or **loss function.** It is a function of θ

$$\mathcal{L}(\theta) = -\sum_{(X,y)\in\mathcal{D}_{\text{train}}} \log P(y|X;\theta)$$

Discriminative Model Objective Function

The discriminative model is parameterized by θ

$$P(y|X;\theta)$$

We often use **negative log likelihood** over training data as our **objective function** or **loss function.** It is a function of θ

$$\mathcal{L}(\theta) = -\sum_{(X,y)\in\mathcal{D}_{\text{train}}} \log P(y|X;\theta)$$

We then optimize the parameters to minimize the loss. Better model has lower loss

$$\hat{\theta} = \arg\min_{\theta} \mathcal{L}(\theta)$$

How to Learn the parameters

- optimization

Optimize Objective Function by Gradient Descent

Calculate the gradient of the loss function with respect to the parameter

$$\frac{\partial \mathcal{L}(\theta)}{\partial \theta}$$

Update θ by moving a small step in the gradient direction to decrease the loss

$$\theta_{\text{new}} \leftarrow \theta_{\text{old}} - \eta \frac{\partial \mathcal{L}(\theta)}{\partial \theta}$$

η is the **learning rate**

Gradient Descent

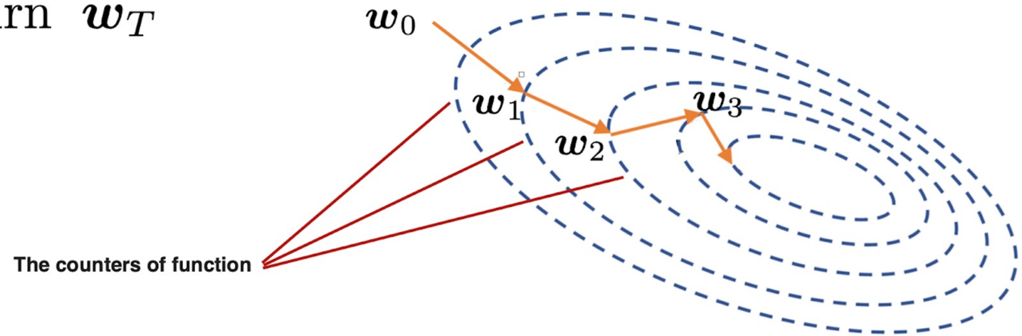
$$\boldsymbol{w}_0 = \mathbf{0}$$

for $\quad t = 1, 2, \ldots, T$

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta_t \nabla f(\boldsymbol{w}_t)$$

end for

return $\;\boldsymbol{w}_T$



Evaluation

- **Text classification**
- Ranking
- Natural language generation (NLG)

Evaluation

Accuracy

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

		Predicted condition	
	Total population = P + N	Positive (PP)	Negative (PN)
Actual condition	Positive (P)	True positive (TP), hit	False negative (FN), type II error, miss, underestimation
	Negative (N)	False positive (FP), type I error, false alarm, overestimation	True negative (TN), correct rejection

https://en.wikipedia.org/wiki/Evaluation_of_binary_classifiers

Evaluation

Precision

$$\text{Precision} = \frac{tp}{tp + fp}$$

Recall

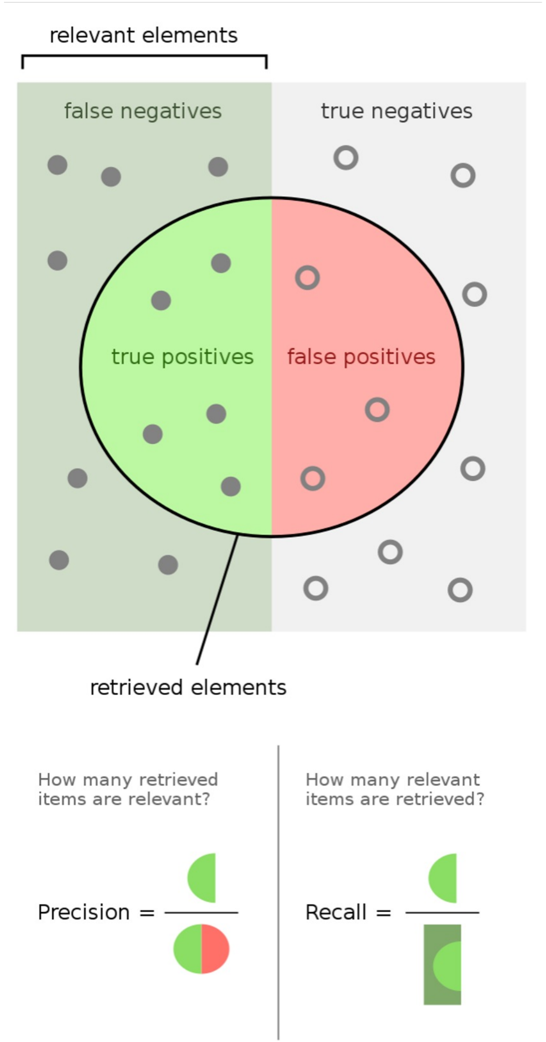
$$\text{Recall} = \frac{tp}{tp + fn}$$

F1 score: the harmonic mean of precision and recall

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

- harmonic mean < geometric mean < arithmetic mean

https://en.wikipedia.org/wiki/Precision_and_recall



Evaluation (F-1 score)

- The following confusion matrix summarizes the predictions made by the model:

		Predicted	
		Drafted = Yes	**Drafted = No**
Actual	**Drafted = Yes**	**120** (True Positive)	**40** (False Negative)
	Drafted = No	**70** (False positive)	**170** (True Negative)

- Precision, Recall?

Evaluation (F-1 score)

- The following confusion matrix summarizes the predictions made by the model:

		Predicted	
		Drafted = Yes	**Drafted = No**
Actual	**Drafted = Yes**	**120** (True Positive)	**40** (False Negative)
	Drafted = No	**70** (False positive)	**170** (True Negative)

- Precision, Recall, and F-1?

F-1 v.s. acc

- For example, suppose 90% of reviews are positive. If we have a model that simply predicts every review to be positive, the model would have a 90% acc.

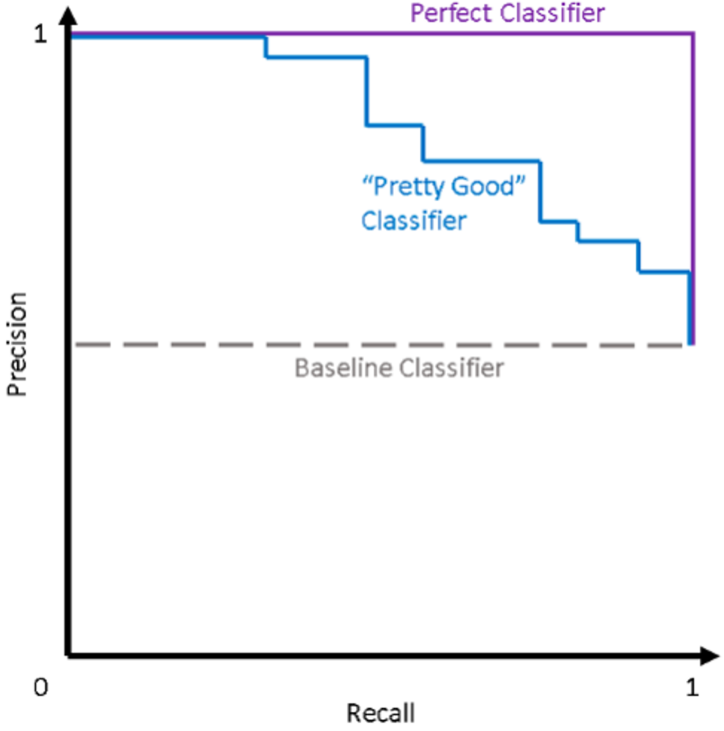
- Value seems high, but ...

Rule of thumb

- We often use **accuracy** when the classes are balanced and there is no major downside to predicting false negatives.

- We often use **F1 score** when the classes are imbalanced and there is a serious downside to predicting false negatives.

Precision-Recall Curve



https://medium.com/@douglaspsteen/precision-recall-curves-d32e5b290248

ROC Curve

Recall is also called **True Positive Rate**

- True Positive / (True Positive + False Negative)

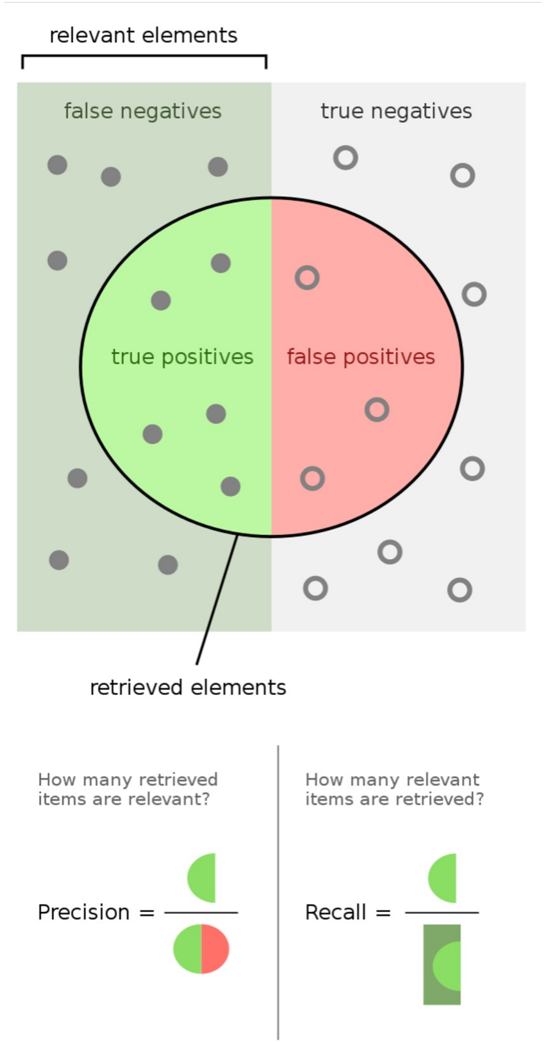
We can also define **False Positive Rate**

- False Positive / (False Positive + True Negative)

Be default, we can use 0.5 as threshold, but we can use other threshold as well. (varying)

As we change the threshold, both TPR and FPR change.

ROC curve: Receiver Operating Characteristic



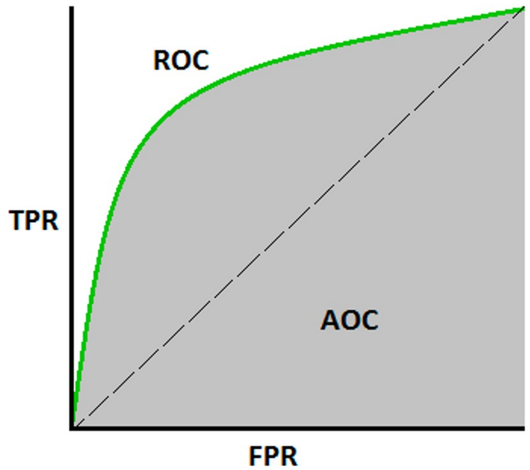
ROC Curve

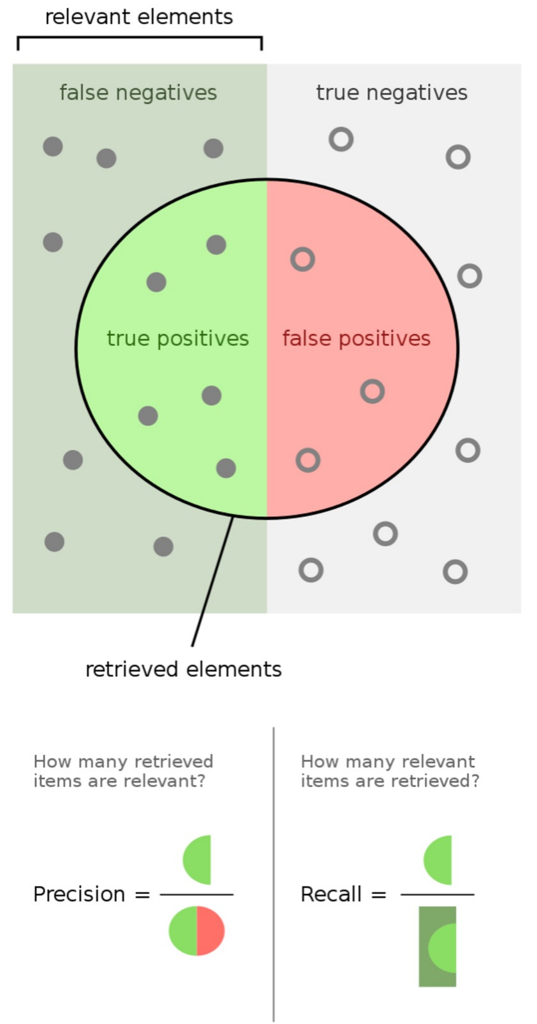
Recall is also called **True Positive Rate**

- True Positive / (True Positive + False Negative)

We can also define **False Positive Rate**

- False Positive / (False Positive + True Negative)





ROC is a probability curve

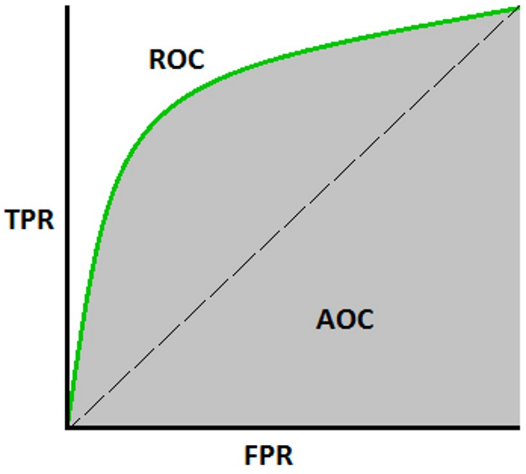
AUC represents the degree or measure of separability

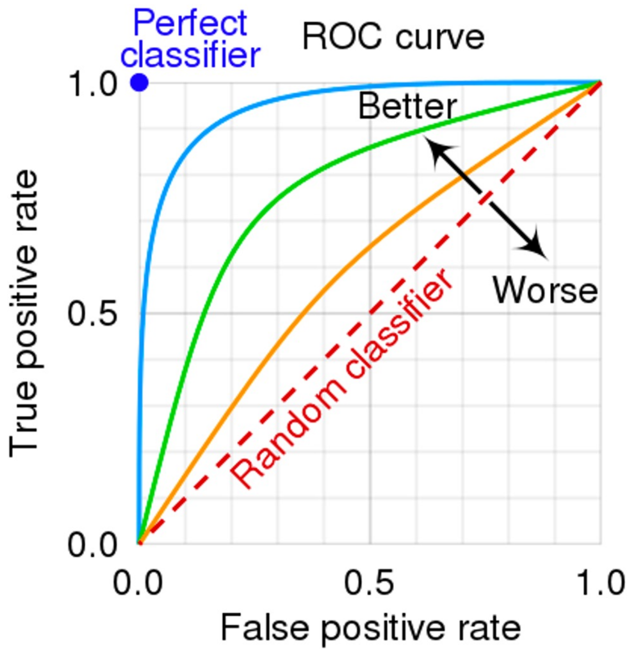
https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

AUC

AUC: Area under Curve

- AUC = 1: The perfect classifier
- AUC = 0.5: The random classifier
- AUC = 0: The worst classifier





Evaluation

- Text classification
- **Ranking (e.g. Information Retrieval / Question Answering)**
- Natural language generation (NLG)

Evaluation for Ranking

Classification: order of predictions doesn't matter

Ranking: order of predictions does matter

- Search engines: Predict which documents match a query on a search engine.

k	Document ID	Predicted Relevance	Actual Relevance
1	06	0.90	Relevant (1.0)
2	03	0.85	Not Relevant (0.0)
3	05	0.71	Relevant (1.0)
4	00	0.63	Relevant (1.0)
5	04	0.47	Not Relevant (0.0)
6	02	0.36	Relevant (1.0)
7	01	0.24	Not Relevant (0.0)
8	07	0.16	Not Relevant (0.0)

https://queirozf.com/entries/evaluation-metrics-for-ranking-problems-introduction-and-examples

Evaluation for Ranking (P@K)

Classification: order of predictions doesn't matter

Ranking: order of predictions does matter

- Search engines: Predict which documents match a query on a search engine.

$$\text{Precision@}k = \frac{true\ positives\ @k}{(true\ positives\ @k) + (false\ positives\ @k)}$$

k	Document ID	Predicted Relevance	Actual Relevance
1	06	0.90	Relevant (1.0)
2	03	0.85	Not Relevant (0.0)
3	05	0.71	Relevant (1.0)
4	00	0.63	Relevant (1.0)
5	04	0.47	Not Relevant (0.0)
6	02	0.36	Relevant (1.0)
7	01	0.24	Not Relevant (0.0)
8	07	0.16	Not Relevant (0.0)

https://queirozf.com/entries/evaluation-metrics-for-ranking-problems-introduction-and-examples

Evaluation for Ranking (Precision@K)

$$\text{Precision@}k = \frac{\textit{true positives @k}}{(\textit{true positives @k}) + (\textit{false positives @k})}$$

k	Document ID	Predicted Relevance	Actual Relevance
1	06	0.90	Relevant (1.0)
2	03	0.85	Not Relevant (0.0)
3	05	0.71	Relevant (1.0)
4	00	0.63	Relevant (1.0)
5	04	0.47	Not Relevant (0.0)
6	02	0.36	Relevant (1.0)
7	01	0.24	Not Relevant (0.0)
8	07	0.16	Not Relevant (0.0)

P@1 = ?

P@4 = ?

P@8 = ?

https://queirozf.com/entries/evaluation-metrics-for-ranking-problems-introduction-and-examples

Evaluation for Ranking (Recall@K)

$$\text{Recall@}k = \frac{\textit{true positives @}k}{(\textit{true positives @}k) + (\textit{false negatives @}k)}$$

R@1 = ?

R@4 = ?

R@8 = ?

k	Document ID	Predicted Relevance	Actual Relevance
1	06	0.90	Relevant (1.0)
2	03	0.85	Not Relevant (0.0)
3	05	0.71	Relevant (1.0)
4	00	0.63	Relevant (1.0)
5	04	0.47	Not Relevant (0.0)
6	02	0.36	Relevant (1.0)
7	01	0.24	Not Relevant (0.0)
8	07	0.16	Not Relevant (0.0)

https://queirozf.com/entries/evaluation-metrics-for-ranking-problems-introduction-and-examples

Evaluation for Ranking (F1@K)

$$F_1@k = 2 \cdot \frac{(Precision@k) \cdot (Recall@k)}{(Precision@k) + (Recall@k)}$$

k	Document ID	Predicted Relevance	Actual Relevance
1	06	0.90	Relevant (1.0)
2	03	0.85	Not Relevant (0.0)
3	05	0.71	Relevant (1.0)
4	00	0.63	Relevant (1.0)
5	04	0.47	Not Relevant (0.0)
6	02	0.36	Relevant (1.0)
7	01	0.24	Not Relevant (0.0)
8	07	0.16	Not Relevant (0.0)

https://queirozf.com/entries/evaluation-metrics-for-ranking-problems-introduction-and-examples

Evaluation for Ranking

Mean Average Precision

- average of AP over all examples in a test set.

There are many metrics for ranking.

- DCG/NDCG: the document relevance is a real number, not simple 0 or 1.

k	Document ID	Predicted Relevance	Actual Relevance	DCG @k
1	06	0.90	Relevant (1.0)	1.0
2	03	0.85	Not Relevant (0.0)	1.0
3	05	0.71	Relevant (1.0)	1.5
4	00	0.63	Relevant (1.0)	1.93
5	04	0.47	Not Relevant (0.0)	1.93
6	02	0.36	Relevant (1.0)	2.29
7	01	0.24	Not Relevant (0.0)	2.29
8	07	0.16	Not Relevant (0.0)	2.29

Evaluation for Ranking

There are many metrics for ranking.

- DCG/NDCG: the document relevance is a real number, not simple 0 or 1.
- MAP (Mean Average Precision)
- MRR (Mean reciprocal rank)

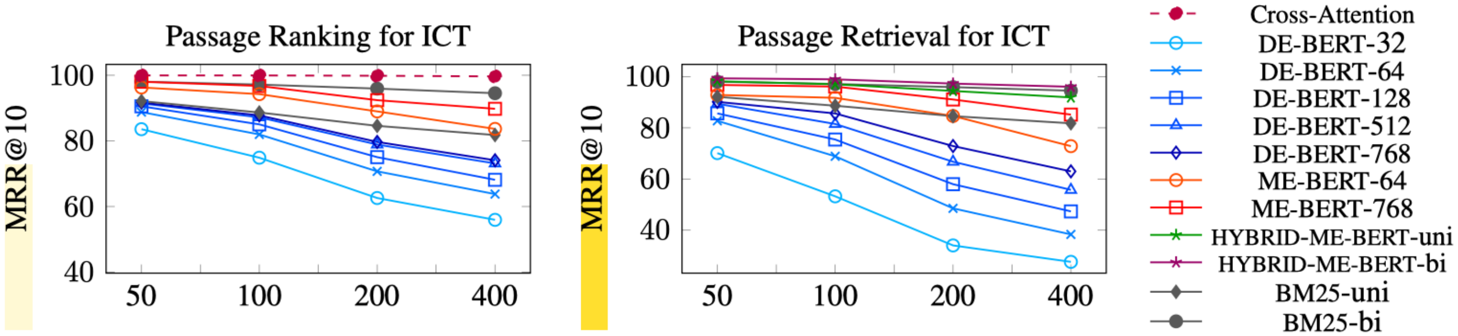


Figure 4: Results on the containing passage ICT task as maximum passage length varies (50 to 400 tokens). *Left*: Reranking 200 candidates; *Right*: Retrieval from three million candidates.

Evaluation for Natural Language **Generation**

(We will talk about this in more detail when we have lectures on NLG and Summarization.)

Automatic Evaluation

n-gram based

- Machine Translation: BLEU
- Summarization: ROUGE, METEOR
- Embedding-based Metric: MoverScore, BERTScore

🌟 Human Evaluation

Is Automatic Metric really reliable? How to determine

Evaluation for Natural Language **Generation**

(We will talk about this in more detail when we have lectures on NLG and Summarization.)

Automatic Evaluation

- Machine Translation: BLEU
- Summarization: ROUGE, METEOR
- Embedding-based Metric: MoverScore, BERTScore

🌟 Human Evaluation

Is Automatic Metric really reliable? Correlation Analysis

Statistical Testing

We have two models ("baseline and our model") with similar accuracies. How can we tell whether the differences are due to consistent trends that hold on other datasets?

	Dataset 1	Dataset 2	Dataset 3
Generative	**0.854**	**0.915**	0.567
Discriminative	0.853	0.902	**0.570**

We need perform **Statistical (significance) testing**!

See The Hitchhiker's Guide to Testing Statistical Significance in Natural Language Processing (Dror et al., ACL 2018) for a complete overview.

Slides Credit: Graham Neubig

Significance Testing: Basic Idea

Given a quantity, we test certain values of uncertainty with respect to the quantity, e.g.

- **p-value**: what is the probability that a difference with another quantity is by chance (lower = more likelihood of a **significant** difference).
- **confidence interval**: what is the range under which we could expect another trial to fall?

Slides Credit: Graham Neubig

Unpaired vs. Paired Tests

Unpaired Test: Compare means of a quantity on two unrelated groups

- Example: test significance of difference of accuracies of a model on two datasets

Paired Test: Compare means of a quantity on one dataset under two conditions

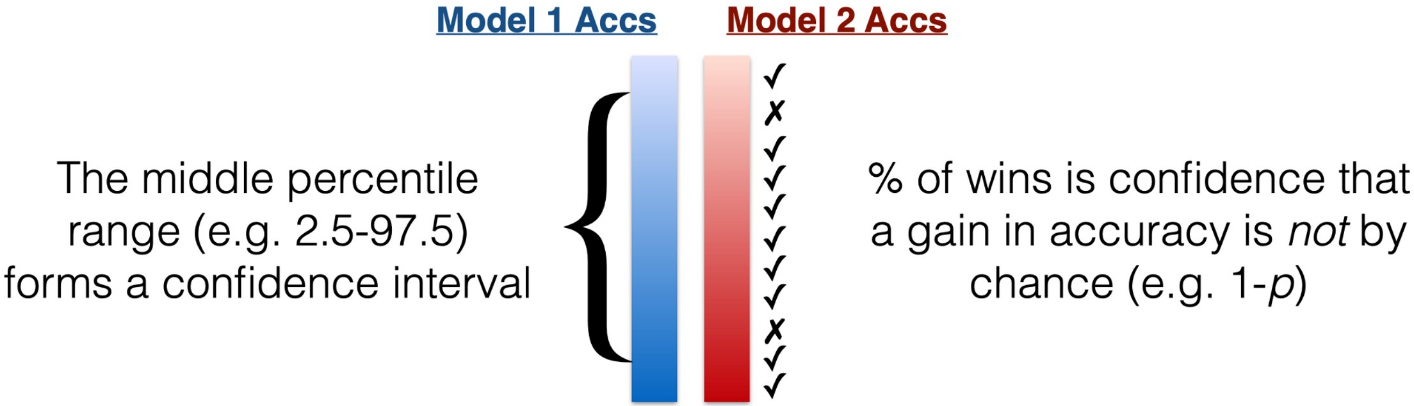
- Example: test significance of difference of accuracies of two models on one dataset

We are most commonly interested in **Paired Test**!

Slides Credit: Graham Neubig

Bootstrap Tests

A method that can measure p-values, confidence intervals, etc. by re-sampling data.
Sample many (e.g. 10,000) subsets from your dev/test set with replacement.
Measure accuracies on these many subsets.
Easy to implement, applicable to any evaluation measure, but somewhat biased on small datasets.



Slides Credit: Graham Neubig

Reporting results with Bootstrap Tests

- Our model outperforms "significantly" ......

Model	Event Trigger Identification P	R	F1	Event Trigger Classification P	R	F1	Event Argument Identification P	R	F1	Argument Role Classification P	R	F1
JOINTBEAM (Li et al., 2013)	76.6	58.7	66.5	74.0	56.7	64.2	74.6	25.5	38.0	68.8	23.5	35.0
STAGEDMAXENT	73.9	**66.5**	70.0	70.4	63.3	66.7	**75.7**	20.2	31.9	**71.2**	19.0	30.0
WITHINEVENT	76.9	63.8	69.7	74.7	62.0	67.7	72.4	37.2	49.2	69.9	35.9	47.4
JOINTEVENTENTITY	**77.6**	65.4	**71.0***	**75.1**	63.3	**68.7**	73.7	**38.5**	**50.6***	70.6	**36.9**	**48.4***

Table 3: Event extraction results on the ACE2005 test set. $*$ indicates that the difference in F1 compared to the second best model (WITHINEVENT) is statistically significant ($p < 0.05$).

[10] All significance tests reported in this paper were computed using the paired bootstrap procedure (Berg-Kirkpatrick et al., 2012) with 10,000 samples of the test documents.

Joint Extraction of Events and Entities within a Document Context (Yang and Mitchell, 2016)